05

Clase Gerente

Transcripción

[00:00] Vamos a ver aquí esta presentación. Bueno, aquí tenemos nuestra clase funcionario que acabamos de crear. Nuestro funcionario pues va a tener su nombre, su documento y su salario tal como lo hemos definido, y además de eso va a tener una bonificación de 10%. ¿Qué quiere decir?

[00:21] Que nuestro funcionario va a necesitar un método que le calcule su bonificación basado en un 10% de su salario. Volvemos al código. ¿Cómo lo implementamos aquí en el lenguaje Java? Entonces vamos a crear aquí nuestro método getBonificación. Perfecto. Abrimos llaves, y como el salario es del tipo double, este también va a ser un public double. Listo.

[00:57] Aquí retornamos. Como la bonificación va a ser el 10% de su salario, vamos a retornar this.salario porque estamos accediendo a la variable salario de esta clase, recuérdalo, por 0.1 para obtener el 10% del salario. Perfecto. Entonces, esto nos va a traer la bonificación del salario del funcionario Diego. Vamos a ver si es verdad.

[01:31] Le vamos a dar aquí a diego.getBonificacion. Ah, aquí olvidé ponerlo dentro de un system.out.printIn, voy a cortar esto, voy a bajar aquí un poco. Acomodamos esto y aquí otra vez un system.out.printIn, "Ctrl + V", ahora sí guardamos. Acá me está dando un pequeño error de compilación. ¿Por qué? Porque no guardé aquí el método aún. Lo guardo aquí, ya no me da el error de compilación.

[02:10] Le damos botón verde y listo. Me da primero mi salario, 2000, y luego me da mi bonificación que en este caso es el 10% de mi salario, que son 200. Igual, hasta ahora nada nuevo en el código, solamente he creado un método más a una clase que ya existe. Esto ya lo hemos visto en el curso anterior y poco a poco vamos a ir evolucionando el código.

[02:39] ¿Qué pasaría si aparte de tener funcionarios yo tengo lo que sería tipo de funcionario? Por ejemplo, en un banco hay un gerente también. Entonces, el gerente va a tener también su nombre, su documento, su salario y también su bonificación. Vamos a la presentación nuevamente para entenderlo mejor.

[03:06] Sabemos que nuestro funcionario tiene su bonificación de 10%. Pero en este caso el gerente por ser gerente, él no va a tener una bonificación de 10%. La bonificación del gerente va a ser un salario completo más. ¿Qué quiere decir? Que el gerente prácticamente como bonificación va a recibir dos veces su salario.

[03:32] ¿Cómo expresamos esto en código Java? Vamos a crear nuevamente otra clase y la vamos a llamar gerente. ¿Qué les parece? Le damos finish y nuevamente vamos a declarar private String nombre, private String documento, private String, perdón, aquí es double salario. ¿No sienten ocmo un pequeño dejá vu?

[04:15] Ya hemos hecho esto antes. ¿De aquí qué sigue? Darle clic derecho, source, generamos getters y setters, generamos los campos. Todo esto es muy repetitivo. ¿No creen? Aquí para darle bonificación voy a crear un public double getBonificacion, cierro, abro llaves, ¿y qué retorno? Retorno this.salario.

[04:50] ¿Por qué? Porque la bonificación es un salario más para el caso del gerente. Guardo aquí y voy a crear otro test aquí también que se va a llamar TestGerente. Exacto. TestGerente. Nada nuevo hasta aquí, voy a crear mi método main, como ya sabemos. Voy a crear un gerente igual new gerente.

Estoy repitiendo exactamente los mismos pasos, así que voy a saltarme aquí de frente, le voy a asignar un salario al Gerente.

[05:37] Gerente.setSalario, su salario va a ser de 5000, y voy a hacer un sout, system.out.printIn de la bonificación. Gerente.getBonificacion, perfecto. Le doy a guardar, ejecuto y perfecto, la bonificación es 5000. Mismo salario. Hasta ahora el código funciona, no hemos visto nada nuevo, pero yo quiero que ustedes detecten algo aquí. No sé si ya lo notaron, ¿pero no se les hace muy parecido todo este código?

[06:16] Los mismos atributos, hasta los mismos métodos, get, set, lo único que varía es getBonificacion, porque él tiene una bonificación y él tiene otra bonificación, entonces quizás podríamos implementar un campo llamado tipo. No sé qué les parece. A mí se me ocurre algo así, porque no me gusta duplicar el código. Entonces por ejemplo ya que tengo dos tipos, un funcionario y un gerente, voy a darle un private int tipo.

[06:55] Y ya que la bonificación se calcula en base a si es gerente o si es funcionario, le puedo decir aquí que si tipo es 1, es gerente, y si tipo es 0, es funcionario. Perfecto. Entonces vamos a ver aquí. Vamos a crearle if abrimos, condición, y aquí this.tipo igual a cero, entonces como es funcionario, va a retornar esto de aquí.

[07:55] Si no, ahí usamos else if (this.tipo) es igual a 1, retornamos this.salario. ¿Qué les parece? Y si no, por defecto, no retornamos nada. ¿Soluciona? Yo creo que soluciona. Vamos a probarlo. Entonces vamos a crearle un getter y setter a tipo. Vamos a nuevamente clic derecho, source, getter y setter a tipo. Generar.

[08:44] Ahí tenemos un getter y setter aquí. Vamos a nuestra clase TestGerente. Llegamos aquí. Y ahora yo le voy a decir que se olvide de esto, voy a comentar esto y voy a crear un Funcionario gerente igual new Funcionario. Perfecto. Punto y coma al final. Y a gerente le voy a asignar un salario de 6000 esta vez, y

como sé que es un gerente le voy a asignar un tipo. SetTipo y Tipo 1 porque es un gerente, tengo que avisarle, ¿cierto?

[09:30] Entonces aquí, gerente, ¿qué bonificación me traerá? Vamos a guardar todo, le damos okay aquí, y me da 6000, que es el tipo correcto aquí, y mi anterior clase, TestFuncionario, ¿será que seguirá funcionando? Voy a decirle aquí que a Diego, como es un funcionario normal, setTipo(0). Perfecto.

[09:58] Guardo nuevamente. Ejecuto el código y el código funciona normal, entonces ¿problema solucionado? Sí, problema solucionado.